PROJECT REPORT

on

Virtual Library Website

(CSE 4Semester Mini project)

2022-2023

Submitted to:

Mr. Akash Chauhan

(CC-CSE-A-4-Sem)

Submitted by:

Mr. Abhishek Rana

Roll. No.: 2118099

CSE-A-4-Sem

Session: 2022-2023

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

GRAPHIC ERA HILL UNVERSITY, DEHRADUN

Graphic Era Hill University
DEHRADUN • BHIMTAL • HALDWANI

CANDIDATE'S DECLARATION

I hereby certify that the work which is being presented in the project report entitled **"Virtual Library Website"** in partial fulfillment of the requirements for the award of the Degree of Bachelor of Technology in Computer Science and Engineering of the Graphic Era Hill University, Dehradun shall be carried out by myself under the mentorship of **Mr.Akash Chauhan, Assistant Professor**, Department of Computer Science and Engineering, Graphic Era Hill University, Dehradun.

Name: Abhishek Rana **University Roll no.: 2118099**

ACKNOWLEDGMENT

We would like to express our gratitude to The Almighty Shiva Baba, the most Beneficent and the most Merciful, for completion of project.

We wish to thank our parents for their continuing support and encouragement. We also wish to thank them for providing us with the opportunity to reach this far in our studies.

We would like to thank particularly our Class Co-ordinator Mr. Akash Chauhan for his patience, support and encouragement throughout the completion of this project and having faith in us.

At last but not the least We greatly indebted to all other persons who directly or indirectly helped us during this work.

Mr. Abhishek Rana

Roll No.- 2118099

CSE-A-IV-Sem

Session: 2022-2023

GEHU, Dehradun

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

1.1 ABOUT PROJECT

I decided to make a Virtual Library Website as I like to read different types of book and I find it difficult to find all the books on a single platform. So through my website, I have tried to promote reading books or reading in general as a hobby.

I had taken a course on ***Web Development with Django*** where I learnt about ***Django*** framework for backend web development.Django is a simple,complete and easy to understand which is most suitable to beginners and for development of good projects.

Using the same I completed my website which has a lot of interesting features. I tried to put all my learning into the project and will be continuing to add more features in the future.

Besides python, I have used ***HTML, CSS, Bootstrap, twilio,*** etc.

1.2 MOTIVATION

The discovery of the printing press marked a pivotal moment in history, triggering a revolution that profoundly transformed various aspects of society. The impact of the printing press can be best understood by examining its effects on knowledge dissemination, cultural shifts, religious reforms, and political transformations.

With the invention of the printing press by Johannes Gutenberg in the mid-15th century, the process of reproducing written material became significantly faster, cheaper, and more efficient. This breakthrough led to a dramatic increase in the availability and accessibility of books, facilitating the widespread dissemination of knowledge. Previously, books were painstakingly copied by hand, making them scarce and expensive. The printing press revolutionized the production of books, enabling their mass production and distribution. As a result, literacy rates rose as more people gained access to printed materials, fostering the spread of knowledge and ideas.

The impact of the printing press was not limited to knowledge dissemination alone; it also sparked cultural shifts and intellectual revolutions. The increased availability of books led to the rise of a literate middle class and the expansion of education. It empowered individuals to engage with a wide range of subjects, explore new ideas, and challenge established beliefs. The Renaissance, a period of remarkable intellectual and artistic achievements, was greatly facilitated by the printing press. Scholars, scientists, and philosophers could now share their works more widely, fostering the exchange of ideas and the advancement of human knowledge.

Religiously, the printing press played a crucial role in the Protestant Reformation. Martin Luther's Ninety-Five Theses, which criticized certain practices of the Catholic Church, were printed and widely circulated, sparking a movement that led to the establishment of Protestantism as a distinct branch of Christianity. The printing press enabled the rapid dissemination of religious texts, allowing people to interpret scripture on their own and challenging the monopoly of religious authority. Religious conflicts and divisions arose as differing interpretations spread, forever altering the religious landscape of Europe.

Politically, the printing press contributed to the spread of revolutionary ideas and the rise of nationalism. Political pamphlets, manifestos, and revolutionary texts were printed and distributed, galvanizing public opinion and fueling movements for social and political

change. The American and French Revolutions, for instance, were greatly influenced by the dissemination of revolutionary literature. The printing press provided a platform for the voices of dissent, enabling the circulation of ideas that questioned traditional power structures and advocated for greater individual rights and liberties.

In summary, the discovery of the printing press revolutionized society by democratizing access to knowledge, fostering cultural shifts, igniting religious reforms, and contributing to political transformations. It empowered individuals, sparked intellectual revolutions, and paved the way for the spread of ideas that continue to shape our world today. The printing press stands as one of the most influential inventions in human history, marking a turning point in the way information is shared, ideas are exchanged, and societies evolve.

Books have been man’s best friend throughout centuries, but in the modern times, the interests of humans have shifted to other platforms like movies, series, social media, etc. These things are not at all bad but have replaced books to some extent.

So to reinitiate the book revolution, I created this website so that people can get all types of books at one platform without much trouble and they can share knowledge and information just like the old times.

CHAPTER 2
PROJECT

2.1 LIBRARIES USED

1. **`django.shortcuts`**: This module provides various shortcuts for common tasks in Django, such as rendering templates, performing redirects, and getting objects from the database.

2. **`django.http`**: This module contains classes and functions related to handling HTTP requests and responses in Django. The `HttpResponse` class represents an HTTP response that can be returned from a view.

3. **`django.contrib.auth.models`**: This module includes the default user model and authentication-related models in Django. The `User` model represents a user account, and the `auth` module provides authentication functions.

4. **`django.contrib`**: This module is part of Django's contributed applications and contains various features and utilities that can be added to a Django project.

5. **`.models`**: This represents a relative import, and it refers to models defined in the current Django application. In the code snippet, it imports specific models such as `Profile`, `Post`, `Comments`, `Books`, and `Reviews`.

6. **`django.contrib.auth.decorators`**: This module provides decorators for controlling access to views based on authentication status. The `login_required` decorator is used to restrict access to views only to authenticated users.

7. **`random`**: This module provides functions for generating random numbers, selecting random elements from sequences, and other randomization-related operations.

8. `**twilio.rest**`: This library allows integration with the Twilio communication platform. The `Client` class from this library provides an interface to interact with various Twilio services like sending SMS, making phone calls, etc.

9. '**django.db.models'**: This module provides classes and functions for defining and working with database models in Django. It includes the Model class, which is the base class for creating models that represent database tables.

10. '**django.contrib.auth'**: This module provides authentication-related functionalities in Django. It includes functions, classes, and models for managing user authentication, permissions, and user sessions.

11. '**Get_user_model':** This function returns the currently active user model class. It's used to retrieve the user model that is currently being used in the Django project, which may be a custom user model defined by the developer.

12. '**django.urls.path'**: This is the same path function mentioned earlier. It is used for defining URL patterns in Django's URL configuration.

These libraries and modules are commonly used in Django web applications for handling HTTP requests, rendering templates, managing authentication, working with databases, and integrating with external services like Twilio.

2.2 FEATURES OF THE WEBSITE

1. **LOGIN-**The users can create accounts on the website and enjoy a lot of benefits

2. **BOOKS-**The website consists of different kinds of books. All these books are available for download in pdf format for free. Every book has been accompanied by reviews from Users of the website, who also can contribute to the website by apploading the books

3. **CATEGORIES OF BOOKS –** The books are organized well according to various categories like fiction, non-fiction, romance, mystery, etc. When uploaded, the books are automatically displayed on the website and sorted according to the category.

4. **REVIEWS-** The users can give their opinions on the book in the form of reviews and read other peoples reviews before downloading a book.

5. **BLOGS-** The website provides users the facility of writing blogs on virtually any topic under the Sun, read others blog and comment on them.

6. **COMMENTS-** The users after reading the blogs can add their comments on the blogs in real time.

7. **SEARCH** – The users can search for a book or a blog on the website. The website will provide him a list of matched books and blogs .

8. **UPLOAD-** The users can also share their books and blogs on the website which will be updated on our website in real time.

9. **CONTACT-** All the users can contact the owner of the website and they will get responses soon after that.

2.3 DATABASE USED

The project has used SQLite database for storing its data.

What is SQLite?

SQLite is a lightweight, serverless, open-source relational database management system (RDBMS) that is widely used in various applications and platforms. It is designed to be embedded within applications and requires minimal setup and administration. SQLite stores the entire database in a single file, making it portable and easy to manage.

Key Features

- Embedded Database
- Relational Database
- Zero Configuration
- Cross Platform Compatibilty
- Small Footprint
- Wide Language Support
- Popular Usage

2.4 ALGORITHMS USED

2.4.1 FOR LOGIN

1. If it's a POST request:
   - Authenticate the user using the provided username and password.
   - If authentication succeeds, log in the user and redirect to the root URL.
   - If authentication fails, create a new user, log them in, create a profile, and redirect to the 'index' URL.
2. If it's not a POST request, render the 'login.html' template.

2.4.2 FOR ADDING A BLOG OR BOOK

For adding a Blog:

1. If it's a POST request, create a new Post object using the provided data and save it to the database. Then, redirect to the root URL.
2. If it's not a POST request, render the 'add-blog.html' template.

For adding a Book:

1. If it's a POST request and the user is logged in, create a new Books object using the provided data and save it to the database. Then, redirect to the root URL.
2. If it's not a POST request or the user is not logged in, render the 'add-book.html' template.

2.4.3 GET A BLOG OR BOOK

1. `post` function:
   - Retrieve the `Post` object with the provided `id` from the database.
   - Retrieve the comments associated with the post.
   - Create a new comment if it's a POST request, using the provided data and linking it to the post. Then, save the comment.

- Render the 'blog-single.html' template with the post and comments as context.

2. `getbook` function:
- Retrieve the `Books` object with the provided `title` from the database.
- Retrieve the reviews associated with the book.
- Create a new review if it's a POST request, using the provided data and linking it to the book. Then, save the review.
- Render the 'get-book.html' template with the book and reviews as context.

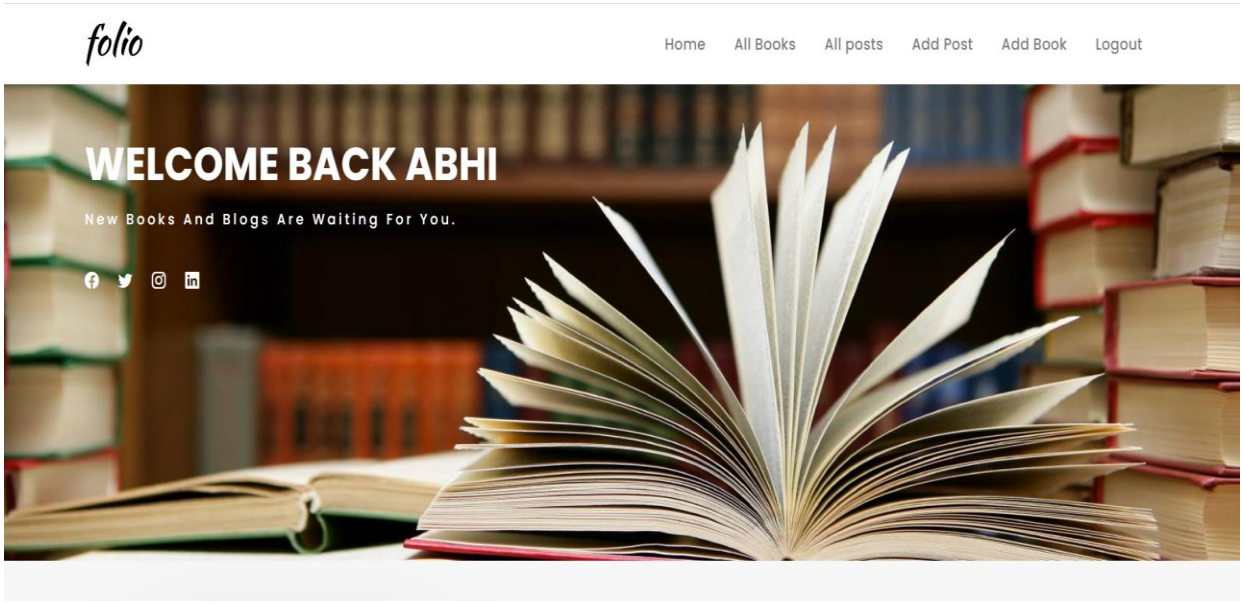
2.4.4 CONTACT SECTION

1. If it's a POST request:
- Retrieve the name, email, and message from the request's POST data.
- Create a `Client` object using the provided Twilio account SID and authentication token.
- Use the `Client` object to send a text message, including the name, email, and message in the message body. Set the sender and recipient phone numbers.
- Print the message SID for reference.
- Redirect to the 'thanks' URL.

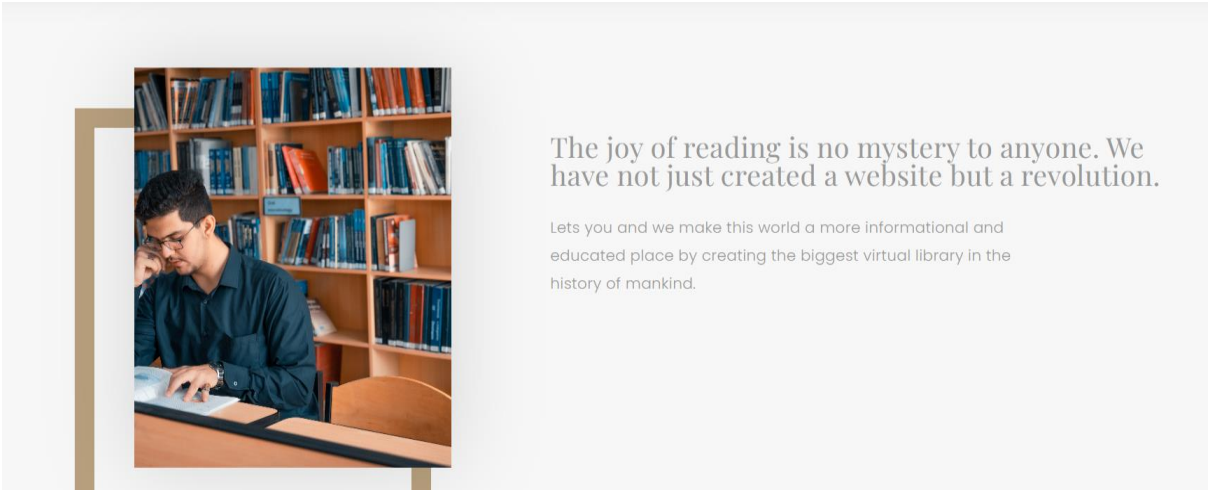
CHAPTER 3

SNAPSHOT OF PROJECT

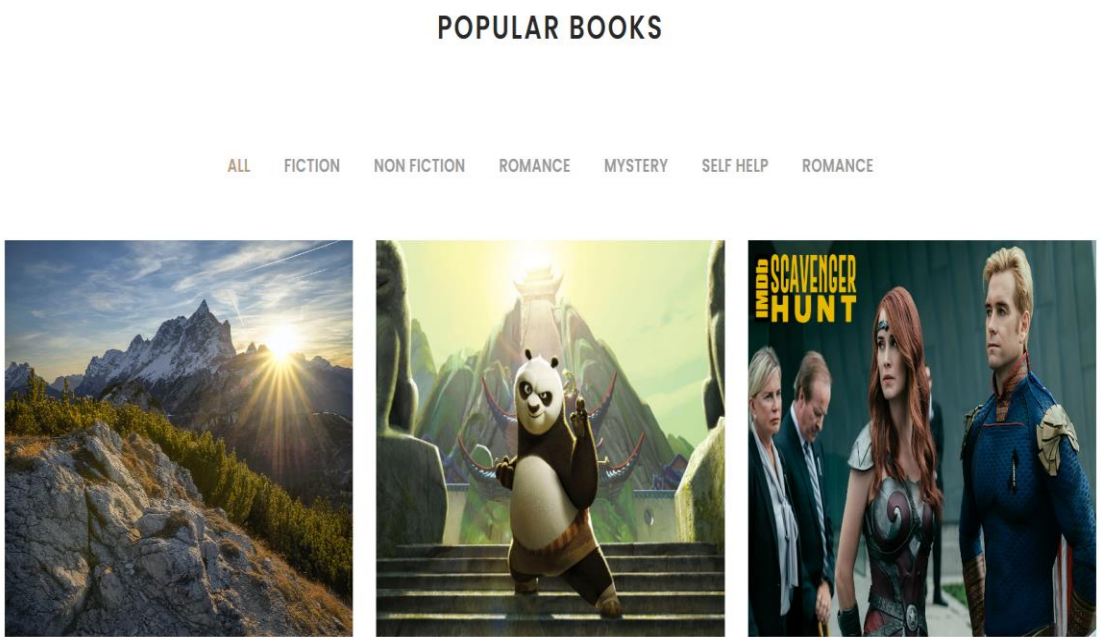
3.1 Header and Top Section



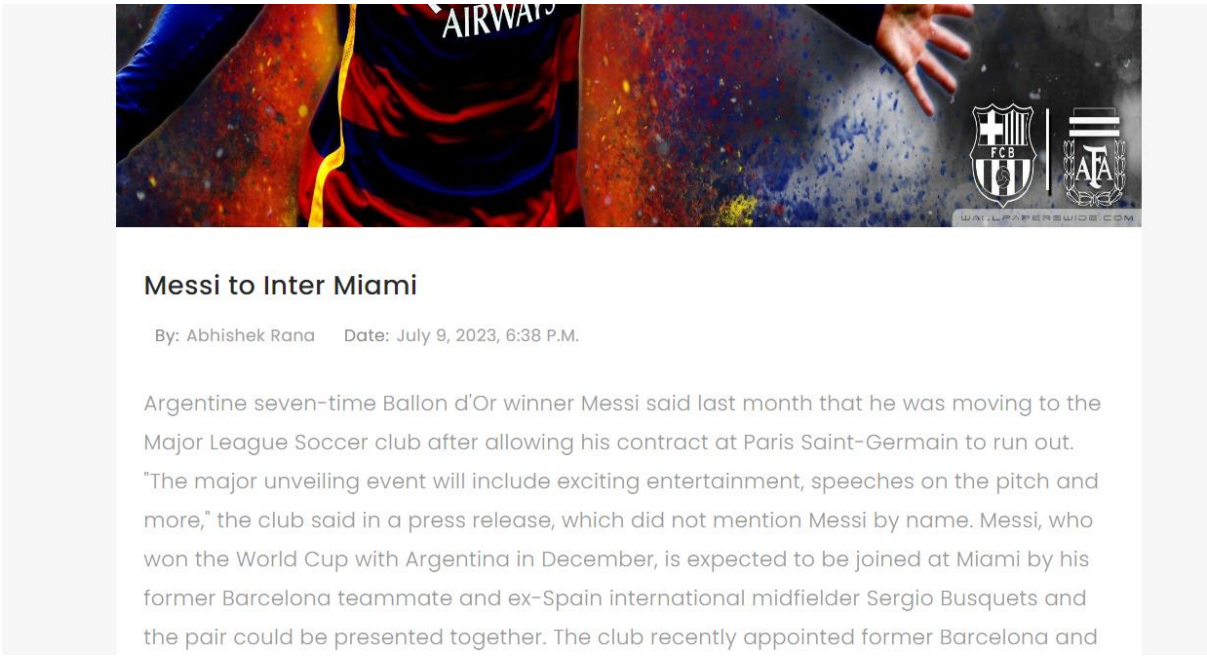
3.2 Our Mission Section



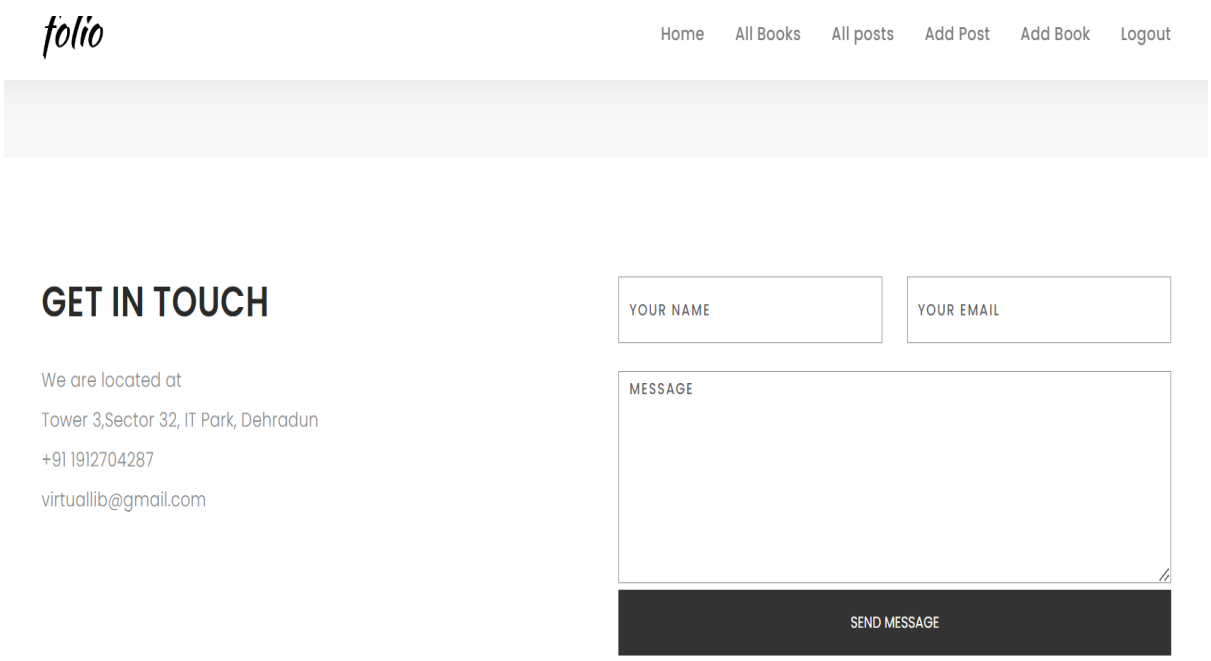
3.3 All Books and Blogs Section



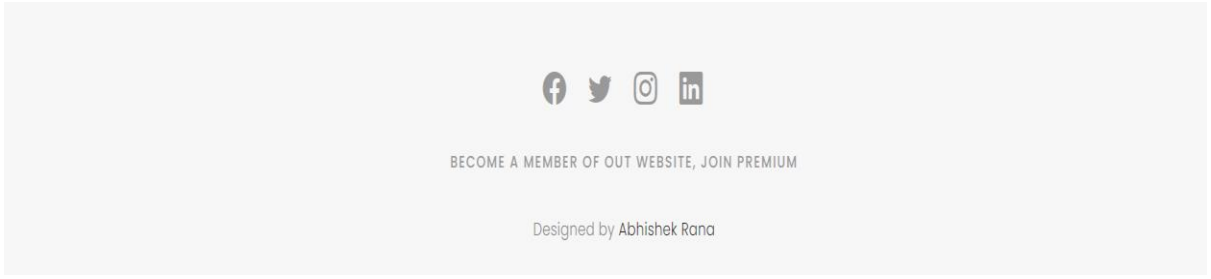
3.4 Blog Section



3.5. Contact Form



3.6. Footer



CHAPTER 4

CONCLUSION

4.1 SUMMARY

Using my website, users can

- Login and Create Accouts
- Download and Read Books
- Upload Books
- Give Reviews
- Read Blogs
- Write Blogs
- Write Comments
- Search for Blogs and Books
- Contact the owners

4.2 FUTURE SCOPE

This website has the potential of becoming the largest virtual library on the Internet and be a popular platform for millions of book enthusiasts and bloggers.

APPENDIX

Code

```
# Name- Abhishek Rana
# Btech CSE 4 Sem Sec A
# Roll no- 2118099/03

# Source code from 3 different backend .py files
#View.py file code
from django.shortcuts import render,redirect, get_object_or_404
from django.http import HttpResponse
from django.contrib.auth.models import User, auth
from django.contrib import messages
from .models import Profile,Post,Comments,Books, Reviews
from django.contrib.auth.decorators import login_required
import random
from twilio.rest import Client


# Create your views here.
@login_required(login_url='login')
def index(request):
    posts=Post.objects.all()
    books=Books.objects.all()
    context={
        'username':request.user.username,
        'posts':posts,
        'books':books
    }
    return render(request, 'index.html',context)

def login(request):
    if request.method=="POST":
        username=request.POST["username"]
        email=request.POST["email"]
        password=request.POST["password"]
        user= auth.authenticate(username=username,password=password)

        if user is not None:
            auth.login(request,user)
            return redirect('/')
        else:
            user=User.objects.create_user(username=username,
email=email,password=password)
```

```
            user.save()

            #log user in and redirect to settings page
            user_login=auth.authenticate(username=username,password=password)
            auth.login(request,user_login)

            #Create a profile object for new user
            user_model=User.objects.get(username=username)

new_profile=Profile.objects.create(user=user_model,id_user=user_model.id)
            new_profile.save()
            return redirect('index')

    else:
        return render(request,'login.html')

@login_required(login_url='login')
def addblog(request):
    if request.method=="POST":
        name=request.POST["name"]
        title=request.POST["title"]
        subtitle=request.POST["subtitle"]
        content=request.POST["content"]
        image=request.FILES.get('image')
        new_post=Post.objects.create(user=name, image=image, title=title,
subtitle=subtitle, content=content)
        new_post.save()
        return redirect('/')
    else:
        return render(request,'add-blog.html')

@login_required(login_url='login')
def addbook(request):
    if request.method=="POST":
        name=request.POST["name"]
        title=request.POST["title"]
        image=request.FILES.get('image')
        category=request.POST['category']
        author=request.POST['author']
        about=request.POST["about"]
        book=request.FILES.get('book')
        new_book=Books.objects.create(name=name,category=category,
image=image, title=title, author=author, about=about, book=book)
        new_book.save()
        return redirect('/')

    else:
        return render(request,'add-book.html')
```

```
@login_required(login_url='login')
def allposts(request):
    posts=Post.objects.all()
    context={
        'posts':posts
    }
    return render(request,'blog-grid.html',context)

@login_required(login_url='login')
def post(request,id):
    post=Post.objects.filter(id=id)[0]
    comments=Comments.objects.filter(post=post)
    context={
        'post': post,
        'comments':comments
    }
    if request.method=="POST":
        name=request.POST['name']
        comment=request.POST['comment']

new_comment=Comments.objects.create(post=post,name=name,comment=comment)
        new_comment.save()

    return render(request,'blog-single.html',context)

@login_required(login_url='login')
def getbook(request,title):
    book=Books.objects.filter(title=title)[0]
    reviews=Reviews.objects.filter(book=book)
    context={
        'book':book,
        'reviews':reviews
    }
    if(request.method=="POST"):
        name=request.POST['name']
        review=request.POST['review']
        new_review=Reviews.objects.create(name=name, comment=review,book=book)
        new_review.save()
    return render(request,'get-book.html',context)

@login_required(login_url='login')
def download(request,title):
    book=Books.objects.filter(title=title)[0]
    if(request.method=="POST"):
        pdf_file = get_object_or_404(Books, title=title)
        response = HttpResponse(pdf_file.book, content_type='application/pdf')
```

```
        response['Content-Disposition'] = f'attachment;
filename="{pdf_file.title}.pdf"'
        return response
    return redirect('/')


@login_required(login_url='login')
def allbooks(request):
    books=Books.objects.all()
    context={
        'books':books
    }
    return render(request,'all-books.html',context)


@login_required(login_url='login')
def logout(request):
    auth.logout(request)
    return redirect('login')

def contact(request):
    account_sid='AC7f538cde22899d3e4520523ed21164aa'
    auth_token="443e389f144466d402445f2cdef7c2d6"
    if request.method=="POST":
        name = request.POST['name']
        email= request.POST['email']
        message= request.POST['message']
        client = Client(account_sid, auth_token)
        message = client.messages \
            .create(
            body=f"{name}\n{email}\n{message}\n",
            from_='+12345401800',
            to='+917037632200'
        )
        print(message.sid)
        return redirect('thanks')

def thanks(request):
    context={
        'username':request.user.username
    }
    return render(request,'thanks.html')
```

```python
#Urls.py code
from django.urls import path
from .import views

urlpatterns = [
    path('',views.index,name="index"),
    path('login',views.login, name='login'),
    path('add-blog',views.addblog,name='add-blog'),
    path('allposts',views.allposts,name='allposts'),
    path('post/<id>',views.post, name='post'),
    path('add-book',views.addbook,name='add-book'),
    path('get-book/<str:title>',views.getbook,name='get-book'),
    path('download-book/<str:title>',views.download,name='download-book'),
    path('all-books',views.allbooks,name='all-books'),
    path('logout',views.logout,name='logout'),
    path('contact',views.contact,name='contact'),
    path('thanks',views.thanks,name='thanks')
]
```

```python
# Models.py Code
from django.db import models
from django.contrib.auth import get_user_model
import uuid
from datetime import datetime


User=get_user_model()

class Profile(models.Model):
    user = models.ForeignKey(User, on_delete=models.CASCADE)
    id_user=models.IntegerField()
    bio=models.TextField(blank=True)
    profileimg = models.ImageField(upload_to='profile_images',default='book-
icon.png')
    location= models.CharField(max_length=100,blank=True)

    def __str__(self):
        return self.user.username

class Post(models.Model):
    id=models.UUIDField(primary_key=True, default=uuid.uuid4)
    user=models.CharField(max_length=100)
    image=models.ImageField(upload_to="post_images")
    title=models.CharField(unique=True,max_length=100)
    subtitle=models.CharField(max_length=200)
    content=models.TextField(blank=False)
    created_at=models.DateTimeField(default=datetime.now)

    def __str__(self):
        return self.title

class Comments(models.Model):
    post=models.ForeignKey(Post, on_delete=models.CASCADE)
    name=models.CharField(max_length=50)
    comment=models.TextField(blank=False)
    date=models.DateField(default=datetime.now)

    def __str__(self):
        return self.name

class Books(models.Model):
    name = models.CharField(max_length=255)
    title=models.CharField(max_length=100)
    author=models.CharField(max_length=100)
    book = models.FileField(upload_to='books')
    image= models.ImageField(upload_to='books')
```

```
    category=models.CharField(max_length=100)
    about=models.TextField(blank=False)
    uploaded_at = models.DateTimeField(default=datetime.now)

    def __str__(self):
        return self.title

class Reviews(models.Model):
    book=models.ForeignKey(Books, on_delete=models.CASCADE)
    name=models.CharField(max_length=50)
    comment=models.TextField(blank=False)
    date=models.DateField(default=datetime.now)

    def __str__(self):
        return self.name
```

REFERENCE

1. https://www.udemy.com/course/python-django-the-practical-guide/
2. https://www.geeksforgeeks.org/
3. https://www.w3schools.com/
4. https://www.twilio.com/
5. https://flask.palletsprojects.com/
6. https:// www.python.org/
7. https://chat.openai.com/
8. https://bootstrapmade.com/
9. Youtube